Redis Dasar

Eko Kurniawan Khannedy

Eko Kurniawan Khannedy

- Technical architect at one of the biggest ecommerce company in Indonesia
- 12+ years experiences
- www.programmerzamannow.com
- youtube.com/c/ProgrammerZamanNow

Eko Kurniawan Khannedy

- Telegram : @khannedy
- Linkedin : https://www.linkedin.com/company/programmer-zaman-now/
- Facebook : fb.com/ProgrammerZamanNow
- Instagram : instagram.com/programmerzamannow
- Youtube : youtube.com/c/ProgrammerZamanNow
- Telegram Channel : t.me/ProgrammerZamanNow
- Tiktok : https://tiktok.com/@programmerzamannow
- Email : echo.khannedy@gmail.com

Sebelum Belajar

- Mengerti Database Relational (misal MySQL atau PostgreSQL)
- Mengerti cara menggunakan terminal / command prompt
- Mengerti Docker sangat direkomendasikan

Pengenalan Redis

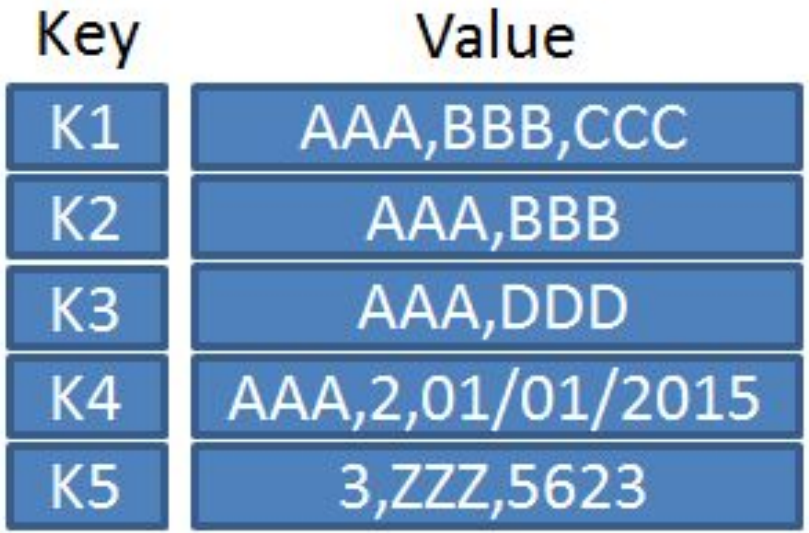
Sejarah Redis

- Redis singkatan dari Remote Dictionary Server adalah sistem basis data key-value berbasis memory
- Pertama kali rilis tahun 2009 sebagai project open source
- https://redis.io/

Apa Itu Key-Value Database?

- Redis adalah sistem basis data berbasis key-value
- Paradigma key-value adalah paradigma dimana data disimpan dalam bentuk pair (key-value)
- Key mirip dengan primary key dari data, sedangkan value adalah isi dari datanya

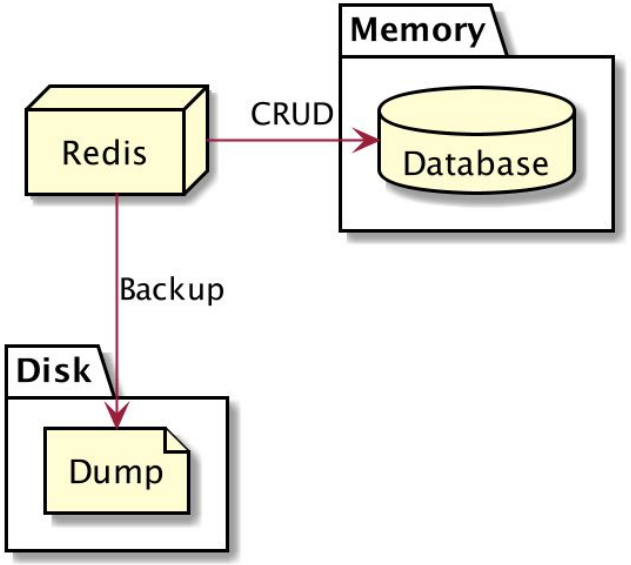
Key-Value Database



Apa Itu In-Memory Database?

- Redis menyimpan datanya di memory, namun kita bisa memintanya untuk menyimpan datanya secara regular permanen di disk.
- Data di disk hanya dijadikan backup ketika redis berjalan ulang, selama redis berjalan, redis hanya akan melakukan manipulasi data ke memory

In-Memory Database



db-engines.com/en/ranking/key-value+store

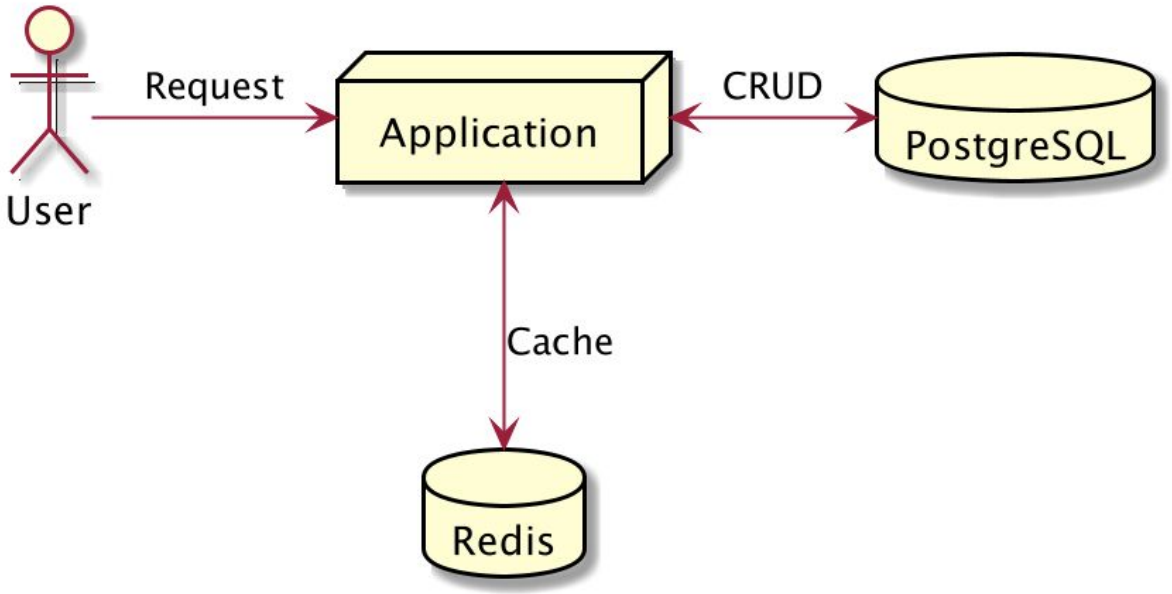
Rank			DBMS	Database Model	Score		
May 2020	Apr 2020	May 2019			May 2020	Apr 2020	May 2019
1.	1.	1.	Redis	Key-value, Multi-model	143.48	-1.33	-4.93
2.	2.	2.	Amazon DynamoDB	Multi-model	64.72	+0.45	+8.78
3.	3.	↑ 4.	Microsoft Azure Cosmos DB	Multi-model	30.68	-1.37	+3.08
4.	4.	↓ 3.	Memcached	Key-value	23.93	-1.41	-4.97
5.	5.	5.	Hazelcast	Key-value, Multi-model	8.67	-0.49	+0.25
6.	6.		etcd	Key-value	7.28	+0.10	
7.	↑ 8.	↑ 8.	Ehcache	Key-value	6.23	-0.03	-0.36
8.	↓ 7.	↓ 6.	Aerospike	Key-value, Multi-model	6.06	-0.88	-1.11
9.	9.	↓ 7.	Riak KV	Key-value	4.76	-0.48	-2.04
10.	↑ 11.	↑ 11.	ArangoDB	Multi-model	4.68	-0.20	-0.11

Kapan Butuh Redis?

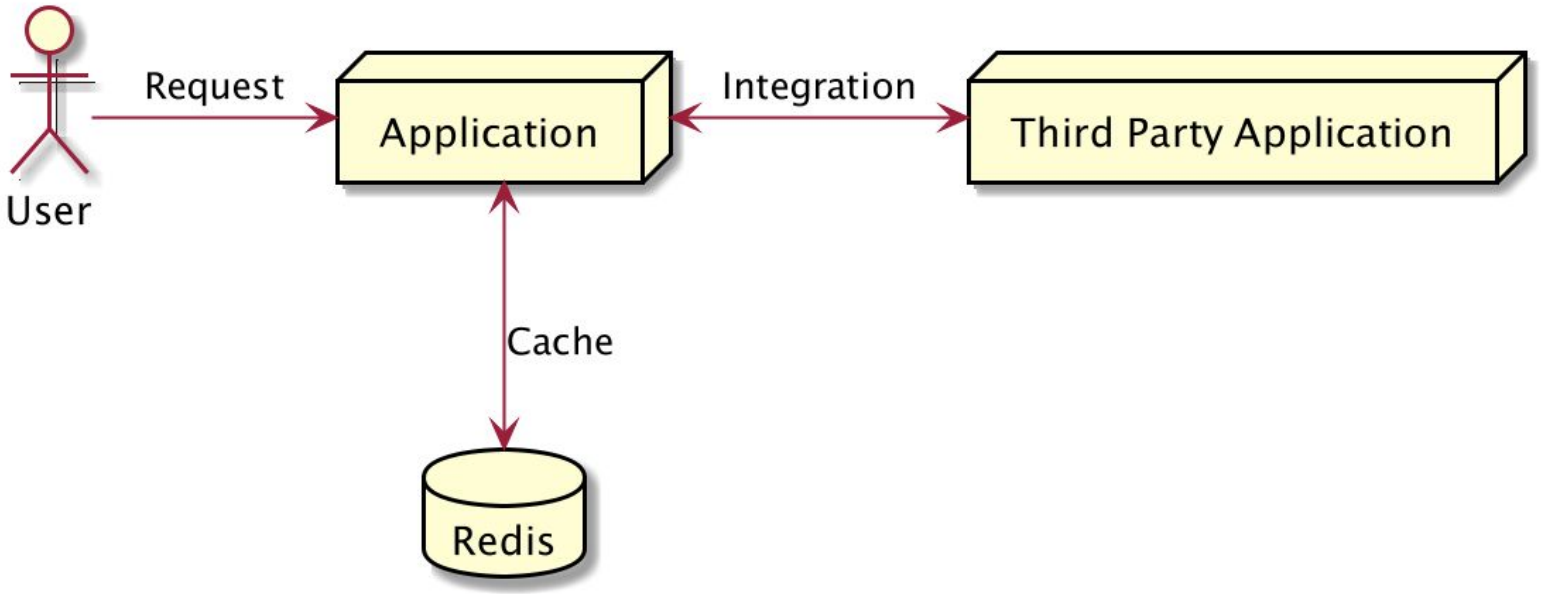
Kapan Butuh Redis?

- Saat kita membuat aplikasi, tidak langsung wajib menggunakan Redis
- Redis menggunakan memory sebagai media penyimpanan utama, otomatis harga memory lebih mahal dibandingkan disk
- Untuk menggunakan Redis, kita perlu lihat kasusnya secara detail

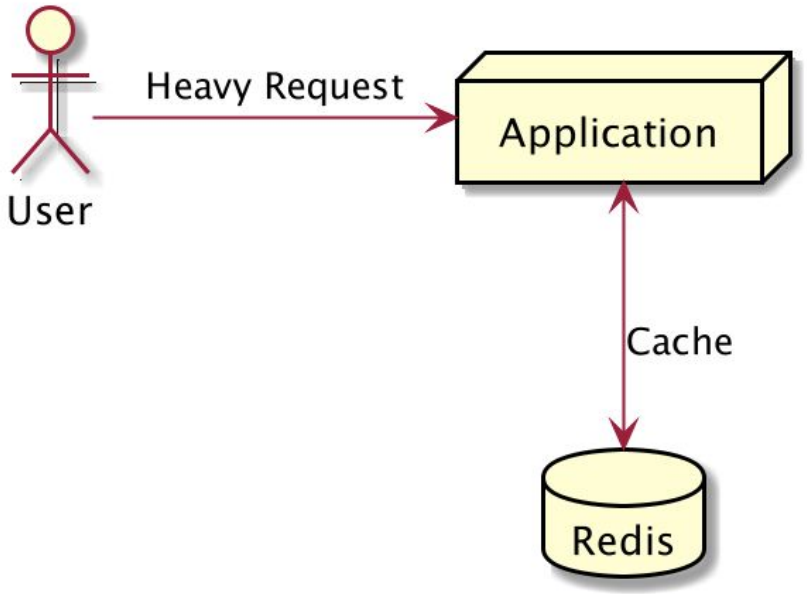
Ketika Database Utama Lambat



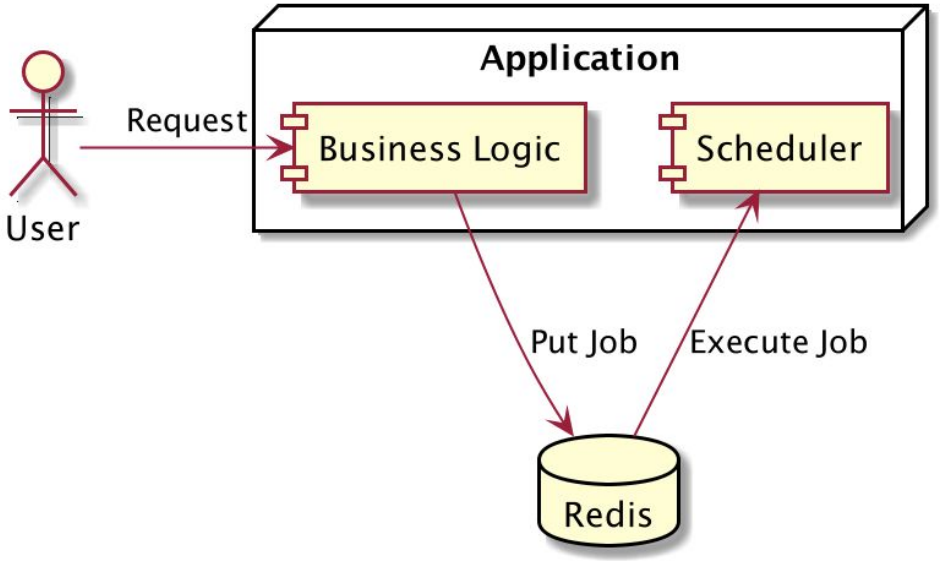
Ketika Aplikasi Lain Lambat



Ketika Ada Proses Berat di Aplikasi



Membuat Delayed Job



Dan masih banyak lainnya

- Rata-rata redis digunakan untuk mempercepat aplikasi yang lambat
- Dan juga redis biasa digunakan untuk caching, menyimpan data secara sementara

Menginstall Redis

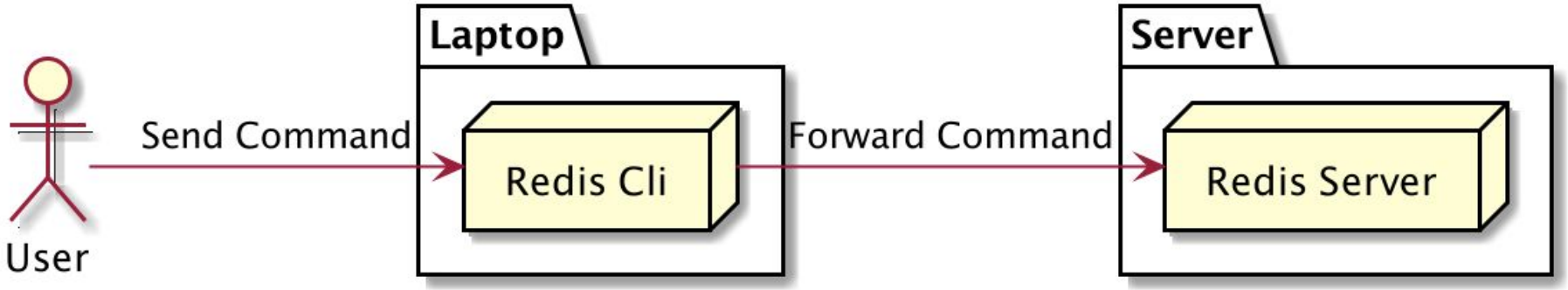
Menginstall Redis

- Redis bisa di install di hampir semua sistem operasi
- Namun cara untuk menginstall di tiap sistem operasi berbeda-beda, kita bisa lihat petunjuk untuk menginstallnya di halaman web Redis https://redis.io/docs/getting-started/installation/
- Windows : https://redis.io/docs/getting-started/installation/install-redis-on-windows/
- Linux : https://redis.io/docs/getting-started/installation/install-redis-on-linux/
- Mac : https://redis.io/docs/getting-started/installation/install-redis-on-mac-os/

Redis Server vs Redis Cli

- Saat kita menginstall Redis, ada 2 aplikasi yang terinstall, Redis Server dan Redis Cli
- Redis Server adalah aplikasi server untuk Redis itu sendiri
- Redis Cli adalah aplikasi command line untuk client, dimana digunakan untuk berkomunikasi dengan Redis Server

Redis Server dan Redis Cli



redis-server.sh

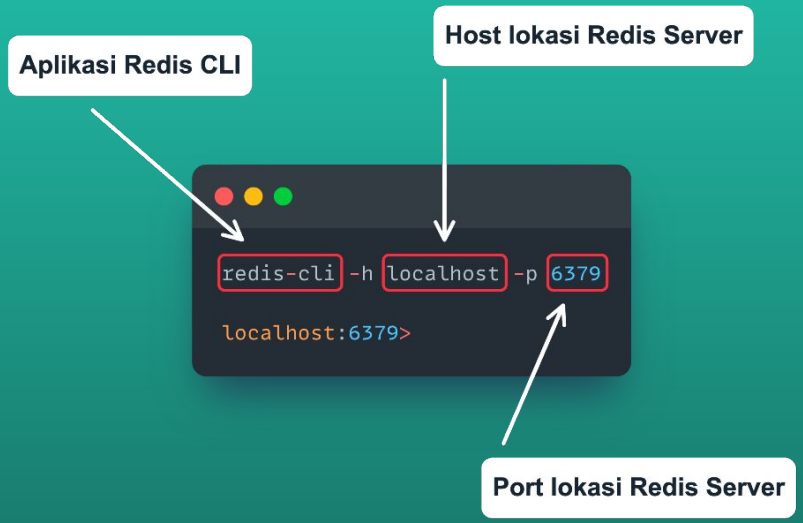
```
redis-server
```

Menjalankan Redis Server

```
26688:C 02 Sep 2023 12:54:22.389 * oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
26688:C 02 Sep 2023 12:54:22.389 * Redis version=7.2.0, bits=64, commit=00000000, modified=0, pid=26688, just started
26688:C 02 Sep 2023 12:54:22.389 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf
26688:M 02 Sep 2023 12:54:22.389 * Increased maximum number of open files to 10032 (it was originally set to 256).
26688:M 02 Sep 2023 12:54:22.389 * monotonic clock: POSIX clock_gettime
                _.-``__ ''-._
           _.-``    `.  `_.  ''-._           Redis 7.2.0 (00000000/0) 64 bit
       .-`` .-```.  ```\/    _.,_ ''-._
      (    '      ,       .-`  | `,    )     Running in standalone mode
      |`-._`-...-` __...-.``-._|'` _.-'|     Port: 6379
      |    `-._   `._    /     _.-'    |     PID: 26688
       `-._    `-._  `-./  _.-'    _.-'
      |`-._`-._    `-.__.-'    _.-'_.-'|
      |    `-._`-._        _.-'_.-'    |           https://redis.io
       `-._    `-._`-.__.-'_.-'    _.-'
      |`-._`-._    `-.__.-'    _.-'_.-'|
      |    `-._`-._        _.-'_.-'    |
       `-._    `-._`-.__.-'_.-'    _.-'
           `-._    `-.__.-'    _.-'
               `-._        _.-'
                   `-.__.-'

26688:M 02 Sep 2023 12:54:22.390 # WARNING: The TCP backlog setting of 511 cannot be enforced because kern.ipc.somaxconn is set to the lower value of 128.
26688:M 02 Sep 2023 12:54:22.391 * Server initialized
26688:M 02 Sep 2023 12:54:22.391 * Ready to accept connections tcp
```

Menjalankan Redis CLI



Configuration

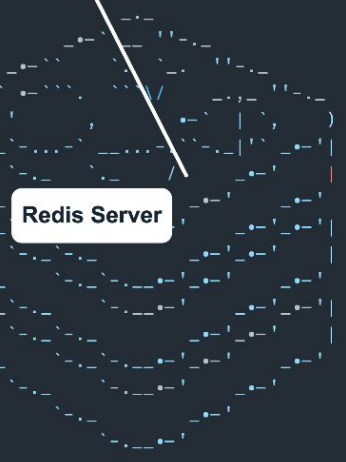
Configuration File

- Saat menjalankan redis, redis tidak butuh file konfigurasi
- Namun jika tidak menggunakan file konfigurasi, redis akan berjalan menggunakan konfigurasi default
- Ada baiknya kita membuat file konfigurasi agar pengaturannya bisa diubah
- https://github.com/redis/redis/blob/7.0/redis.conf

redis-server.sh

```
redis-server config/redis.conf

30279:C 02 Sep 2023 13:09:49.564 * oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
30279:C 02 Sep 2023 13:09:49.564 * Redis version=7.2.0, bits=64, commit=00000000, modified=0, pid=30279, just started
30279:C 02 Sep 2023 13:09:49.564 * Configuration loaded
30279:M 02 Sep 2023 13:09:49.564 * monotonic clock: POSIX clock_gettime
30279:M 02 Sep 2023 13:09:49.565 # Failed to write PID file: Permission denied
```



Redis Server

Lokasi config file

```
                Redis 7.2.0 (00000000/0) 64 bit

                Running in standalone mode
                Port: 6379
                PID: 30279


                     https://redis.io


30279:M 02 Sep 2023 13:09:49.565 # WARNING: The TCP backlog setting of 511 cannot be enforced because kern.ipc.somaxconn is set to the lower value of 128.
30279:M 02 Sep 2023 13:09:49.565 * Server initialized
30279:M 02 Sep 2023 13:09:49.566 * Ready to accept connections tcp
```

Database

Database

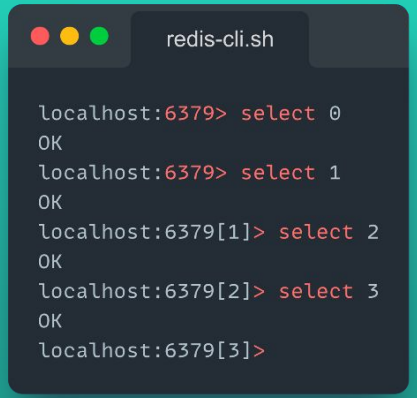
- Redis memiliki konsep database seperti pada relational database mysql atau postgre
- Di redis kita bisa membuat database dan menggunakan database nya
- Namun sedikit berbeda, jika di relational database kita bisa membuat database dengan menggunakan nama database, di redis kita hanya bisa menggunakan angka sebagai database
- Secara default database di redis adalah 0 (nol)
- Kita bisa menggunakan database sejumlah maksimal sesuai dengan konfigurasi yang kita gunakan di file konfigurasi

redis.conf

```
# Set the number of databases. The default database is DB 0, you can select
# a different one on a per-connection basis using SELECT <dbid> where
# dbid is a number between 0 and 'databases'-1
databases 16
```

Operasi Database

Operasi	Keterangan
select database	Selecting database number


redis-cli.sh
localhost:6379> select 0
OK
localhost:6379> select 1
OK
localhost:6379[1]> select 2
OK
localhost:6379[2]> select 3
OK
localhost:6379[3]>

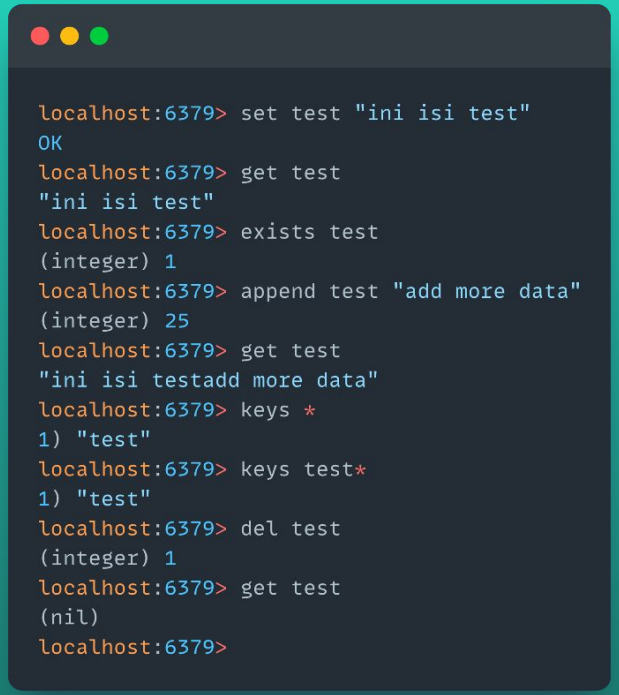
Strings

Struktur Data Redis

- Redis sebenarnya mendukung struktur data yang banyak, seperti String, List, Set, dan lain-lain
- Namun yang paling sering digunakan adalah struktur data String
- Struktur data lainnya akan kita bahas di kelas terpisah, yaitu kelas Redis Data Structure

Operasi Data String

Operasi	Keterangan
set key value	mengubah string value dari key
get key	mendapatkan value menggunakan key
exists key	mengecek apakah key memiliki value
del key [key ...]	menghapus menggunakan key
append key value	menambah data value ke key
keys pattern	mencari key menggunakan patterns


localhost:6379> set test "ini isi test"
OK
localhost:6379> get test
"ini isi test"
localhost:6379> exists test
(integer) 1
localhost:6379> append test "add more data"
(integer) 25
localhost:6379> get test
"ini isi testadd more data"
localhost:6379> keys *
1) "test"
localhost:6379> keys test*
1) "test"
localhost:6379> del test
(integer) 1
localhost:6379> get test
(nil)
localhost:6379>

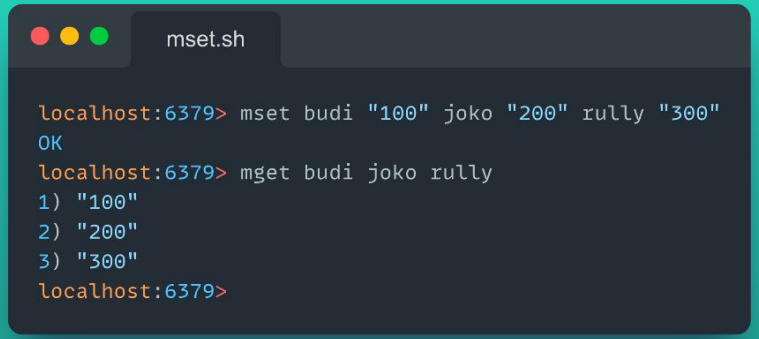
Operasi Range Data String

Operasi	Keterangan
setrange key offset value	mengubah value dari offset yang ditentukan
getrange key start end	mengambil value dari range yang ditentukan


get-range.sh
localhost:6379> set eko "Eko Kurniawan"
OK
localhost:6379> get eko
"Eko Kurniawan"
localhost:6379> setrange eko 4 "Kurniawan Khannedy"
(integer) 22
localhost:6379> get eko
"Eko Kurniawan Khannedy"
localhost:6379> getrange eko 4 12
"Kurniawan"
localhost:6379>

Operasi Multiple Data String

Operasi	Keterangan
mget key [key ...]	Get the values of all the given keys
mset key value [key value ...]	Set multiple keys to multiple values


mset.sh
localhost:6379> mset budi "100" joko "200" rully "300"
OK
localhost:6379> mget budi joko rully
1) "100"
2) "200"
3) "300"
localhost:6379>

Expiration

Expiration

- Secara default saat kita menyimpan data ke redis, redis akan menyimpannya secara permanen sampai kita menghapusnya
- Kadang kita mendapatkan kasus ingin menghapus data di redis secara otomatis dalam waktu tertentu
- Misal kita menyimpan data cache di redis selama 10 menit, setelah 10 menit kita akan query ulang ke database untuk mendapatkan data terbaru
- Hal ini bisa dilakukan di redis, redis memiliki fitur expiration secara otomatis pada data yang kita simpan di redis

Operasi Expiration Data String

Operasi	Keterangan
expire key seconds	Set a key's time to live in seconds
setex key seconds value	Set the value and expiration of a key
ttl key	Get the time to live for a key

ttl.sh

```
localhost:6379> expire eko 5
(integer) 1
localhost:6379> ttl eko
(integer) 3
localhost:6379> get eko
(nil)
localhost:6379> setex eko 10 "Eko Kurniawan Khannedy"
OK
localhost:6379> ttl eko
(integer) 7
localhost:6379> get eko
"Eko Kurniawan Khannedy"
localhost:6379> get eko
(nil)
localhost:6379>
```

Increment & Decrement

Increment & Decrement

- Operasi Increment & Decrement sekilas sangat mudah dilakukan, hanya tinggal mengupdate data yang di redis dengan data baru (data lama ditambah 1)
- Namun jika operasi dilakukan secara paralel dan dalam waktu yang sangat cepat, hal ini bisa memungkinkan race condition
- Untungnya redis memiliki operasi untuk melakukan increment dan decrement

Manual Increment Decrement di Pemrograman

JS race-condition.js

```
let value = await redis.get("key");
value = Number(value) + 1;
await redis.set("key", value);
```

Operasi Increment & Decrement

Operasi	Keterangan
incr key	Increment the integer value of a key by one
decr key	Decrement the integer value of a key by one
incrby key increment	Increment the integer value of a key by the given amount
decrby key decrement	Decrement the integer value of a key by the given number

Increment dan Decrement

increment.sh

```
localhost:6379> incr counter
(integer) 1
localhost:6379> incr counter
(integer) 2
localhost:6379> incr counter
(integer) 3
localhost:6379> get counter
"3"
localhost:6379> decr counter
(integer) 2
localhost:6379> decr counter
(integer) 1
localhost:6379> get counter
"1"
localhost:6379>
```

increment-by.sh

```
localhost:6379> incrby counter 5
(integer) 6
localhost:6379> incrby counter 5
(integer) 11
localhost:6379> incrby counter 5
(integer) 16
localhost:6379> decrby counter 3
(integer) 13
localhost:6379> decrby counter 3
(integer) 10
localhost:6379> decrby counter 3
(integer) 7
localhost:6379> decrby counter 3
(integer) 4
localhost:6379>
```

Flush

Flush

- Kadang kita butuh mengosongkan seluruh data di redis, misal ketika terjadi kesalahan kode sehingga menyebabkan data di redis salah
- Menghapus data di redis satu-satu menggunakan operasi delete bukanlah hal yang bijak
- Redis memiliki fitur untuk menghapus seluruh data di database redis, yaitu operasi flush

Operasi Flush

Operasi	Keterangan
flushdb	Remove all keys from the current database
flushall	Remove all keys from all databases

Flush

flushdb.sh

```
localhost:6379> mget budi joko rully
1) "100"
2) "200"
3) "300"
localhost:6379> flushdb
OK
localhost:6379> mget budi joko rully
1) (nil)
2) (nil)
3) (nil)
localhost:6379>
```

flushall.sh

```
localhost:6379> set eko "Eko"
OK
localhost:6379> select 1
OK
localhost:6379[1]> set eko "Eko"
OK
localhost:6379[1]> flushall
OK
localhost:6379[1]> get eko
(nil)
localhost:6379[1]> select 0
OK
localhost:6379> get eko
(nil)
localhost:6379>
```

Pipeline

Pipeline

- Perintah yang dikirim dari client ke server redis menggunakan Request/Response protocol
- Artinya tiap request yang dikirim ke server redis, maka redis akan membalasnya secara langsung
- Kadang ada kebutuhan kita mengirim data ke redis dalam jumlah besar, misal ketika ada kasus memindahkan data dari database mysql ke redis
- Jika kita mengirim satu per satu datanya, maka akan butuh waktu lama untuk selesai
- Redis mendukung operasi bulk via pipeline, dimana kita bisa mengirim beberapa perintah sekaligus dalam satu request
- Namun perlu diketahui, server redis tidak akan membalas tiap perintah yang dikirim via pipeline

Operasi Pipeline Menggunakan Redis Cli

redis-cli -h host -p port -n database --pipe < input-file

Pipeline

input-file.txt

```
set eko "Eko Kurniawan"
set budi "Budi Nugraha"
set joko "Joko Morro"
set rully "Rully Hidayat"
```

pipeline.sh

```
redis-cli -h localhost -p 6379 -n 0 --pipe < input-file.text
All data transferred. Waiting for the last reply...
Last reply received from server.
errors: 0, replies: 4

redis-cli -h localhost -p 6379
localhost:6379> mget eko budi joko rully
1) "Eko Kurniawan"
2) "Budi Nugraha"
3) "Joko Morro"
4) "Rully Hidayat"
localhost:6379>
```

Database number

Input file

Transaction

Transaction

- Seperti pada database relational, redis juga mendukung transaction
- Proses transaction adalah proses dimana kita mengirimkan beberapa perintah, dan perintah tersebut akan dianggap sukses jika semua perintah sukses, jika gagal maka semua perintah harus dibatalkan

Operasi Transaction

Operasi	Keterangan
multi	Mark the start of a transaction block
exec	Execute all commands issued after MULTI
discard	Discard all commands issued after MULTI

Transaction

transaction.sh

```
localhost:6379> multi
OK
localhost:6379(TX)> set apple "Apple"
QUEUED
localhost:6379(TX)> set samsung "Samsung"
QUEUED
localhost:6379(TX)> set xiaomi "Xiaomi"
QUEUED
localhost:6379(TX)> exec
1) OK
2) OK
3) OK
localhost:6379> mget apple samsung xiaomi
1) "Apple"
2) "Samsung"
3) "Xiaomi"
```

rollback.sh

```
localhost:6379> multi
OK
localhost:6379(TX)> set satu "Satu"
QUEUED
localhost:6379(TX)> set dua "Dua"
QUEUED
localhost:6379(TX)> set tiga "Tiga"
QUEUED
localhost:6379(TX)> discard
OK
localhost:6379> mget satu dua tiga
1) (nil)
2) (nil)
3) (nil)
localhost:6379>
```

Monitor

Monitor

- Kadang ada kasus kita ingin mendebug aplikasi saat berkomunikasi dengan redis
- Redis memiliki fitur monitor, yaitu fitur untuk memonitor semua request yang masuk ke redis server
- Dengan fitur ini kita bisa mudah mendebug jika ternyata ada perintah yang salah yang dikirim oleh aplikasi kita ke redis server

Operasi Monitor

Operasi	Keterangan
monitor	Listen for all requests received by the server in real time

Monitor

client.sh

```
localhost:6379> get eko
"Eko Kurniawan"
localhost:6379> get budi
"Budi Nugraha"
localhost:6379> get joko
"Joko Morro"
localhost:6379>
```

monitor.sh

```
localhost:6379> monitor
OK
1693658852.195965 [0 127.0.0.1:64087] "get" "eko"
1693658853.662835 [0 127.0.0.1:64087] "get" "budi"
1693658855.014258 [0 127.0.0.1:64087] "get" "joko"
```

Server Information

Server Information

- Kadang kita butuh mendapatkan informasi dan statistik redis server
- Seperti jumlah memory yang sudah terpakai, konfigurasi dan lain-lain
- Redis memiliki fitur ini, sehingga kita sangat mudah untuk mendapat informasi server dan memonitor nya

Operasi Server Information

Operasi	Keterangan
info	Get information and statistics about the server
config get <key>	Get the value of a configuration parameter from redis.conf

Server Information

info.sh

```
localhost:6379> info
# Server
redis_version:7.2.0
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:d50c69ff806e6ad2
redis_mode:standalone
os:Darwin 22.3.0 arm64
arch_bits:64
monotonic_clock:POSIX clock_gettime
multiplexing_api:kqueue
atomicvar_api:c11-builtin
gcc_version:4.2.1
process_id:30279
process_supervised:no
run_id:662c6323f0cd14217c49f1b2cce5907e7ce83d58
tcp_port:6379
server_time_usec:1693659128149952
uptime_in_seconds:24139
uptime_in_days:0
hz:10
```

config.sh

```
localhost:6379> config get databases
1) "databases"
2) "16"
localhost:6379> config get bind
1) "bind"
2) "127.0.0.1 -::1"
localhost:6379> config get save
1) "save"
2) "3600 1 300 100 60 10000"
localhost:6379>
```

Client Connection

Client Connection

- Redis menyimpan semua informasi client di server
- Hal ini memudahkan kita untuk melihat daftar client, dan juga mengecek jika ada anomali, seperti terlalu banyak koneksi client ke redis

Operasi Server Information

Operasi	Keterangan
client list	Get the list of client connections
client id	Returns the client ID for the current connection
client kill ip:port	Kill the connection of a client

Client Information

client.sh

```
localhost:6379> client list
id=7 addr=127.0.0.1:64087 laddr=127.0.0.1:6379 fd=8 name= age=620 idle=0
id=8 addr=127.0.0.1:64216 laddr=127.0.0.1:6379 fd=9 name= age=413 idle=126
localhost:6379> client id
(integer) 7
localhost:6379> client kill 127.0.0.1:64216
OK
localhost:6379>
```

Protected Mode

Protected Mode

- Secara default, ketika kita menyalakan redis server, redis server akan mendengarkan request dari semua network interface. Ini sangat berbahaya, karena bisa jadi redis terekspos secara public
- Namun, redis punya second layer untuk pengecekan koneksi, yaitu mode protected, secara default mode protectednya aktif, artinya walaupun redis bisa diakses dari manapun, tapi redis hanya mau menerima request dari 127.0.0.1 (localhost)

Network Configuration

redis.conf

```
# IF YOU ARE SURE YOU WANT YOUR INSTANCE TO LISTEN TO ALL THE INTERFACES
# COMMENT OUT THE FOLLOWING LINE.
#
# You will also need to set a password unless you explicitly disable protected
# mode.
# ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
# bind 127.0.0.1 -::1
bind 192.168.68.153

# By default protected mode is enabled. You should disable it only if
# you are sure you want clients from other hosts to connect to Redis
# even if no authentication is configured.
protected-mode yes
```

Protected Mode

redis-cli.sh

```
redis-cli -h 192.168.68.153 -p 6379 -n 0
192.168.68.153:6379> get eko
(error) DENIED Redis is running in protected mode because protected mode is enabled and no password is set for the default user.
In this mode connections are only accepted from the loopback interface. If you want to connect from external computers to Redis you
may adopt one of the following solutions: 1) Just disable protected mode sending the command 'CONFIG SET protected-mode no' from
the loopback interface by connecting to Redis from the same host the server is running, however MAKE SURE Redis is not publicly
accessible from internet if you do so. Use CONFIG REWRITE to make this change permanent. 2) Alternatively you can just disable
the protected mode by editing the Redis configuration file, and setting the protected mode option to 'no', and then restarting
the server. 3) If you started the server manually just for testing, restart it with the '--protected-mode no' option. 4)
Set up an authentication password for the default user. NOTE: You only need to do one of the above things in order for the
server to start accepting connections from the outside.
192.168.68.153:6379>
```

Security

Authentication

- Authentication adalah proses verifikasi identitas untuk memastikan bahwa yang mengakses adalah identitas yang benar
- Redis memiliki fitur authentication, dan kita bisa menambahkannya di file konfigurasi di server redis
- Namun perlu diingat, proses authentication di redis itu sangat cepat, jadi pastikan gunakan password sepanjang mungkin agar tidak mudah untuk di brute force

Authorization

- Authorization adalah prose memberi hak akses terhadap identitas yang telah berhasil melewati proses authentication
- Redis mendukung hal ini, jadi kita bisa membatasi hak akses apa saja yang bisa dilakukan oleh identitas yang kita buat
- https://redis.io/docs/management/security/acl/

Security

redis.conf

```
# For more information about ACL configuration please refer to
# the Redis web site at https://redis.io/topics/acl

user default on +@connection
user eko on +@all ~* >rahasia
```

Authentication

Operasi	Keterangan
auth <username> <password>	melakukan autentikasi menggunakan username dan password

Authentication

auth.sh

```
redis-cli -h 192.168.68.153 -p 6379 -n 0
192.168.68.153:6379> mget eko budi joko
(error) NOAUTH Authentication required.
192.168.68.153:6379> auth eko rahasia
OK
192.168.68.153:6379> mget eko budi joko
1) "Eko Kurniawan"
2) "Budi Nugraha"
3) "Joko Morro"
192.168.68.153:6379>
```

Persistence

Persistence

- Media penyimpanan utama redis adalah di memory
- Namun kita bisa menyimpan data di memory redis tersebut di disk jika kita mau
- Namun perlu diingat proses penyimpanan data ke disk redis tidak realtime, dia dilakukan secara scheduler dengan konfigurasi tertentu
- Jadi jangan jadikan redis sebagai media penyimpanan persistence, gunakan redis sebagai database untuk membantu database persistence lainnya

Persistence Config

redis.conf

```
# Unless specified otherwise, by default Redis will save the DB:
#   * After 3600 seconds (an hour) if at least 1 change was performed
#   * After 300 seconds (5 minutes) if at least 100 changes were performed
#   * After 60 seconds if at least 10000 changes were performed
#
# You can set these explicitly by uncommenting the following line.
#
save 3600 1 300 100 60 10000
```

Operasi Persistence

Operasi	Keterangan
save	Synchronously save the dataset to disk
bgsave	Asynchronously save the dataset to disk

Eviction

Ketika Memory Redis Penuh

- Ketika memory redis penuh, maka redis secara default akan mereject semua request penyimpanan data
- Hal ini mungkin menjadi masalah ketika kita hanya menggunakan redis sebagai cache untuk media penyimpanan sementara
- Kadang akan sangat berguna jika memory penuh, redis bisa secara otomatis menghapus data yang sudah jarang digunakan

Eviction

- Redis mendukung fitur eviction (menghapus data lama, dan menerima data baru)
- Namun untuk mengaktifkan fitur ini, kita perlu memberi tahu redis, maximum memory yang boleh digunakan, dan bagaimana strategi untuk melakukan eviction nya
- https://redis.io/docs/reference/eviction/

Eviction Config

redis.conf

```
# In short... if you have replicas attached it is suggested that you set a lower
# limit for maxmemory so that there is some free RAM on the system for replica
# output buffers (but this is not needed if the policy is 'noeviction').
#
maxmemory 10mb

# Note: with any of the above policies, when there are no suitable keys for
# eviction, Redis will return an error on write operations that require
# more memory. These are usually commands that create new keys, add data or
# modify existing keys. A few examples are: SET, INCR, HSET, LPUSH, SUNIONSTORE,
# SORT (due to the STORE argument), and EXEC (if the transaction includes any
# command that requires memory).
#
maxmemory-policy noeviction
```

Materi Selanjutnya

Materi Selanjutnya

- Redis Data Structure
- Redis Pubsub
- Redis Replication
- Redis Cluster